



National Certificate of Educational Achievement  
TAUMATA MĀTAURANGA Ā-MOTU KUA TĀEA

## Internal Assessment Resource

### Digital technologies Level 1

This resource supports assessment against:

Achievement Standard 91075 version 2

Construct a plan for a basic computer program for a specified task

**Resource title: Construct a Plan for a Computer Program**

3 credits

Student Name:

.....

*I declare that the material I have submitted for this unit or achievement standard is my own work and that I had no outside help from others in completing it.*

\_\_\_\_\_

*(student to sign)*

Comments:

.....

.....

.....

.....

☐ Not Achieved   ☐ Achieved   ☐ Merit   ☐  
Excellence

DEPARTMENT USE ONLY

Internal Moderation Grade: \_\_\_\_\_ Signed: \_\_\_\_\_ Date: \_\_\_\_\_

*If your grade differs from the mark given by the teacher, fill in the 'internal moderation' report.*

## Student instructions

### Introduction

This assessment task requires you to construct a plan for a computer program. This should be a racing game as described below, or another program that you have discussed with your teacher.

You will be assessed on how skilfully you construct a plan for the program and the efficiency of the plan.

This is an individual task. You will have 6 weeks of in-class and out-of-class time for teaching, learning and assessment.

### Task

Construct a plan for a basic computer program. You may either use the racing game described in Appendix 1, or an idea that has been approved by your teacher. Your plan should be printed and handed in when it is complete. See appendix 2 for an example of a plan (at Achieved/Merit level).

Your plan must have the following:

#### 1. Drawings of each screen you will make for the game

These can be drawn by hand or using an image editor. As well as drawing, describe what happens on each screen.

#### 2. A list of the variables you will use

Include the following information for each variable:

- Name - This needs to be accurate and descriptive E.g. driver\_name
- Type - E.g. number, text, true/false
- Purpose - A brief description of what the variable is for. E.g. The name of the driver

#### 3. A description of the scripts your game will include

Write pseudocode for each of the scripts your game needs to have. Make sure this is detailed enough that the marker can see clearly what each script does.

#### 4. A detailed test plan

Think through what your program should do, and write a plan that can be followed to test it. Make sure you consider what should happen with normal input (e.g. pressing arrow keys to drive around track) and error conditions (e.g. when the car leaves the track).

---

## **Specifications**

---

The planned program must include:

- At least two variables (e.g. speed, lap\_count, driver\_name) storing at least two types of information (e.g. number, true/false, text)
- Assignment of variables (e.g. set the score to 0)
- Accurate and descriptive variable names, including the names of sprites
- Use of predefined actions (e.g. switch to costume, say "You win!")
- User input (e.g. Arrow keys, Mouse clicks)
- Use of a sequence control structure (i.e. several statements that are run after each other)
- Use of a conditional control structure (e.g. if, if/else, repeat until)
- Use of an iterative control structure (e.g. loops such as repeat until or forever)
- A comprehensive plan for testing the program to identify errors.

---

## **Appendix 1: Racing Game Description**

---

If you wish to create a racing game, it must have the following:

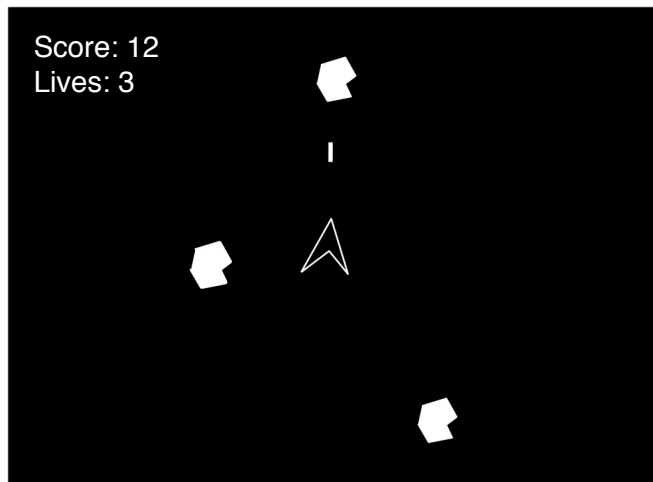
- A start screen with instructions on how to play the game
- A racetrack with at least 3 corners in it (not just a simple oval), a start/finish line, and a car
- Variables of at least two types (e.g. car\_speed, player\_name)
- Reset of the car to the start position behind the start/finish line at the beginning of the race
- Continual forward movement of the car when the race begins
- User control of the car (e.g. using left/right arrow keys to turn and up/down to change the speed)
- Failure to finish the race if the car leaves the track
- A way to change the speed of the car during a race
- Successful completion of the race when the car crosses the finish line at the end of the final lap
- Feedback to the player when the race is over, whether they finished or went off the track (e.g. Game over screen)

---

## Appendix 2: Example plan (Asteroids Game)

---

### Game Screens



This is the main game screen. The ship is in the centre of the screen, and can fire laser bullets at the rocks flying towards it. The rocks appear at the edges of the screen and move toward the centre. The ship cannot move around the screen, but can rotate.



This is the screen you see after you run out of lives from being hit by too many rocks.

### Variables

Name	Type	Purpose
lives	number	Track how many lives the user has left
player_name	text	The player's name. Used for messages.
score	number	The player's score

### Scripts

The ship will have the following scripts:

When flag clicked:

Ask "What is your name?"  
set player\_name to the answer

When left arrow pressed:  
Rotate left by 45 degrees

When right arrow pressed:  
Rotate right by 45 degrees

When space pressed:  
Send message "shoot"

When message "game over" received:  
Say "Well done *player\_name*!" for 3 seconds  
Say "Your final score was *score*" for 3 seconds

**The laser bullet will have the following script:**

When message "shoot" received:  
go to ship position  
point in direction ship is pointing  
show  
repeat until off screen  
    move forward 5 steps  
hide

**Each rock will have the following scripts:**

When message "game over" received:  
hide

When message "game started" received:  
go to random point on edge of screen  
point towards ship  
forever  
    move forward 1 step  
    if hit by laser  
        change costume to exploded  
        increase score by 1  
        wait 1 second  
        change costume to normal  
        go to random point on edge of screen  
        point towards ship  
    else if hit ship  
        decrease lives by 1  
        if lives is 0  
            send message "game over"  
        wait 1 second  
        go to random point on edge of screen  
        point towards ship

**The Stage will have the following scripts:**

When flag clicked:

Switch to costume game\_background  
set score to 0 and lives to 3

When message "game over" received:

Switch to costume game\_over

## Test Plan

Check that:

1. at the beginning of the game the score is 0
2. at the start of the game lives is 3
3. when the game is started, it asks for your name
4. during the game, the left arrow key makes the ship turn left
5. the right arrow key makes the ship turns right
6. when space is pressed, the ship shoots
7. rocks appear at different locations around the edge of the screen
8. visible rocks move towards the ship
9. when a bullet hits a rock, it is destroyed
10. when a rock is destroyed, a new rock appears at the edge of the screen
11. when a rock hits the ship, lives is decreased by 1
12. when lives gets to 0 the game ends
13. at the end of the game you are told what your final score was
14. the final message includes the name you gave at the beginning of the game

## Assessment schedule: Digital Technologies 91075 Construct a Plan for a Computer Program

Evidence/Judgements for Achievement	Evidence/Judgements for Achievement with Merit	Evidence/Judgements for Achievement with Excellence
<p>The student has <b>constructed a plan for a basic computer program</b> for a specified task.</p> <p>The student has, with guidance:</p> <ul style="list-style-type: none"> <li>specified variables and their data types</li> </ul> <p><i>For example the plan includes:</i></p> <ul style="list-style-type: none"> <li><i>driver_name, text</i></li> <li><i>car_speed, number</i></li> <li><i>top_speed, number</i></li> </ul> <ul style="list-style-type: none"> <li>specified a procedural structure that combines actions, conditions and control structures</li> </ul> <p><i>For example:</i></p> <p><i>When the up arrow is pressed, if the car_speed is less than the maximum, increase the car_speed by 1</i></p> <p><i>When the race is started, inside a forever loop, move the car forward by car_speed steps, then if it has left the track send a 'crashed' broadcast.</i></p> <p><i>When the game is started, show the Title screen. When the spacebar is pressed, switch to the race screen and start the race.</i></p> <ul style="list-style-type: none"> <li>specified a set of test cases with expected inputs for testing the program.</li> </ul> <p><i>The student's plan identifies a set of test cases that the student will check to ensure that the game works correctly on expected inputs.</i></p> <p><i>For example:</i></p> <ul style="list-style-type: none"> <li><i>I will check to see that when the left arrow is pressed the car turns left</i></li> <li><i>I will check to see that when the up arrow is pressed the car speeds up</i></li> </ul> <p><i>This description relates to only part of what is required, and is indicative only.</i></p>	<p>The student has <b>skilfully constructed a plan for a basic computer program</b> for a specified task.</p> <p>The student has:</p> <ul style="list-style-type: none"> <li>specified variables and their data types</li> </ul> <p><i>As for achieved</i></p> <ul style="list-style-type: none"> <li><b>independently</b> constructed the plan</li> </ul> <p><i>The student created the plan for their game with limited teacher input and made independent decisions regarding the procedural structure, actions, conditions and control structures to be used within the program.</i></p> <p><i>When making planning decisions, the student referred to previous examples of planning they had completed rather than asking the teacher or peers for assistance.</i></p> <ul style="list-style-type: none"> <li>specified a procedural structure with <b>well-chosen</b> actions, conditions and control structures</li> </ul> <p><i>Planned actions will have the desired effect, without unwanted side effects</i></p> <ul style="list-style-type: none"> <li>specified a set of test cases with expected and boundary inputs for testing the program.</li> </ul> <p><i>For example, the student plans to test:</i></p> <ul style="list-style-type: none"> <li><i>that when the left arrow is pressed the car turns left</i></li> <li><i>that the game functions as expected when the car leaves the track</i></li> <li><i>that the race finishes when the finish line is crossed</i></li> </ul> <p><i>This description relates to only part of what is required, and is indicative only.</i></p>	<p>The student has <b>efficiently constructed a plan for a basic computer program</b> for a specified task.</p> <p>The student has:</p> <ul style="list-style-type: none"> <li>specified variables and their data types</li> </ul> <p><i>As for achieved</i></p> <ul style="list-style-type: none"> <li>independently constructed the plan</li> </ul> <p><i>Same as for Merit</i></p> <ul style="list-style-type: none"> <li>specified a procedural structure with well-chosen actions, conditions and control structures</li> <li>constructed a flexible and robust plan</li> </ul> <p><i>For example:</i></p> <ul style="list-style-type: none"> <li><i>The plan specifies that users can choose how many laps to complete, and counts the laps each time the finish line is passed.</i></li> <li><i>Correct handling of expected, boundary and invalid inputs, or checking input data for validity. This means the program is more efficient to maintain and debug.</i></li> </ul> <ul style="list-style-type: none"> <li>specified an effective procedural structure that constitutes a well-structured logical solution to the task</li> </ul> <p><i>The student has planned a procedural structure for the game that uses control structures with a well-defined purpose and contains no unnecessary duplication or repetition. For example:</i></p>

		<ul style="list-style-type: none"> <li>– <i>The student has planned to use a Boolean variable "canMove" which determines the player's state (in play or crashed into and thus out of play). The plan calls for one script to control all elements of the program which are dependent upon checking the boolean variable "canMove". Thus, "canMove" is only checked only one time every step. (The plan is for a more efficient program because the program doesn't have to check the same variable in three different scripts.)</i></li> <li>• specified a comprehensive set of test cases with expected, boundary and invalid input for testing the program <i>The student's plan specifies tests for expected inputs as well as boundary cases and invalid inputs. For example, the student plans to test:</i> <ul style="list-style-type: none"> <li>– <i>that when the left arrow key is pressed it rotates the car to the left.</i></li> <li>– <i>that the game functions as expected when the car leaves the track</i></li> <li>– <i>that when the space key is pressed after a race is started, it is ignored and the race doesn't restart.</i></li> </ul> </li> </ul> <p><i>This description relates to only part of what is required, and is indicative only.</i></p>
--	--	--

Final grades will be decided using professional judgement based on a holistic examination of the evidence provided against the criteria in the Achievement Standard.